

## FAT-POINTER BASED ON RANGE TLB

Akilan Selvacoumar<sup>1</sup>, Robert Stewart<sup>1</sup>, Hans-Wolfgang Loidl<sup>1</sup>, and Ryad Soobhany<sup>1</sup>

<sup>1</sup>Heriot-Watt University, UK

# Abstract

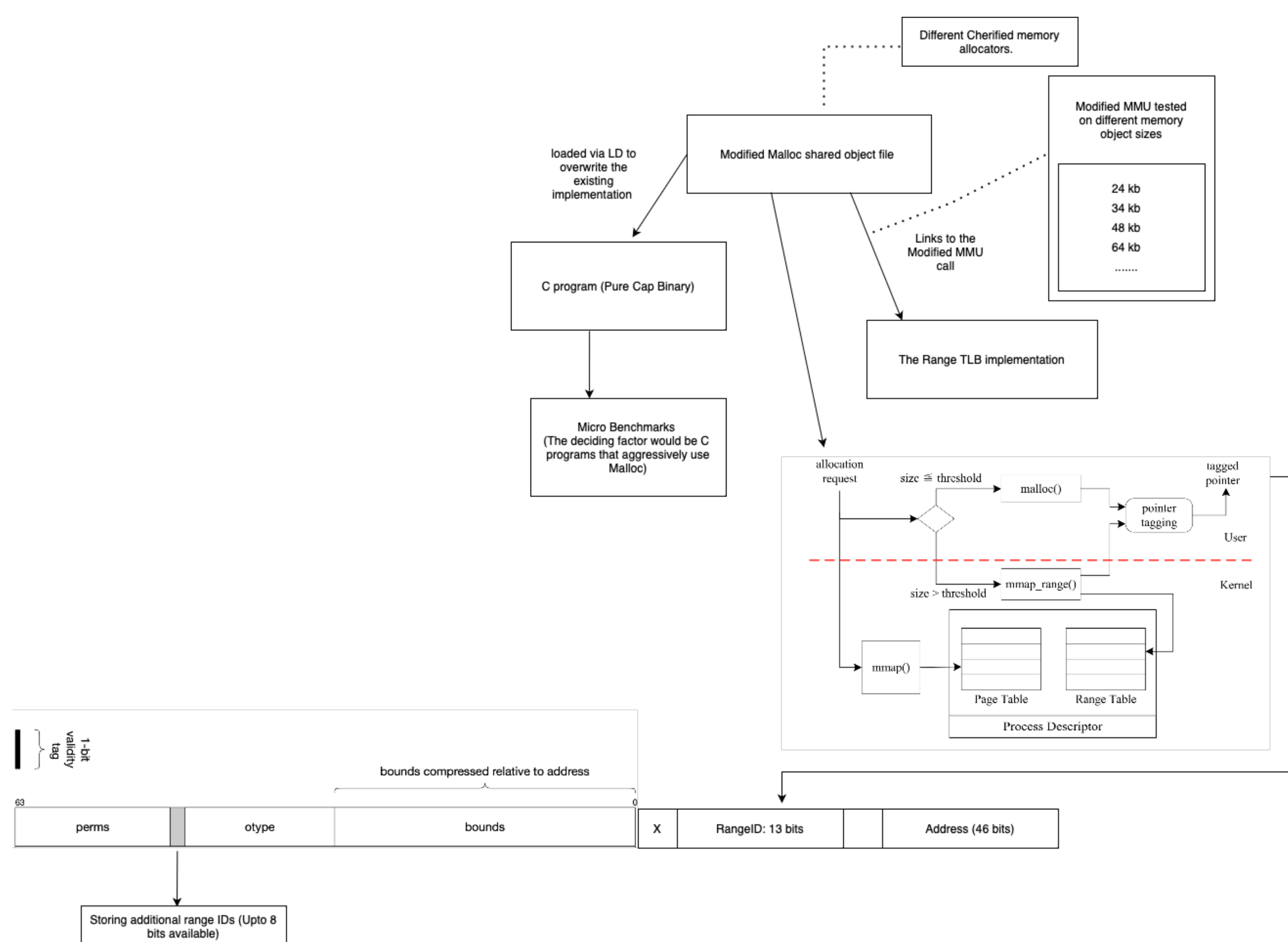
The CHERI architecture is designed to provide security guarantees. This has had performance implications such as more L1 TLB misses on purecap mode rather than running an application as a 64 bit program. The following experiment proposes on adding range TLB which is stored on the page table to store TLB translation entries of large memory objects (This would in turn lookup TLB entries in the range table rather than looking up the L1 TLB entry). During this process the range ID would be added to the pointer which allows looking up the TLB entries at an earlier stage during the pipeline. The FAT pointer structure of CHERI allows to encode custom bits of meta-data which in this case can be used to store the ID of the range TLB entry.

## Research Questions and Goals

*Research Questions:*

- (RQ1) Is there an optimization that explicitly uses FAT pointers to reduce the number of TLB misses to reduce the CPU STALL Time ?
- (RQ2) Can the number of L1 DTLB misses be minimised by offloading translation lookups to the newly introduced range TLB ?

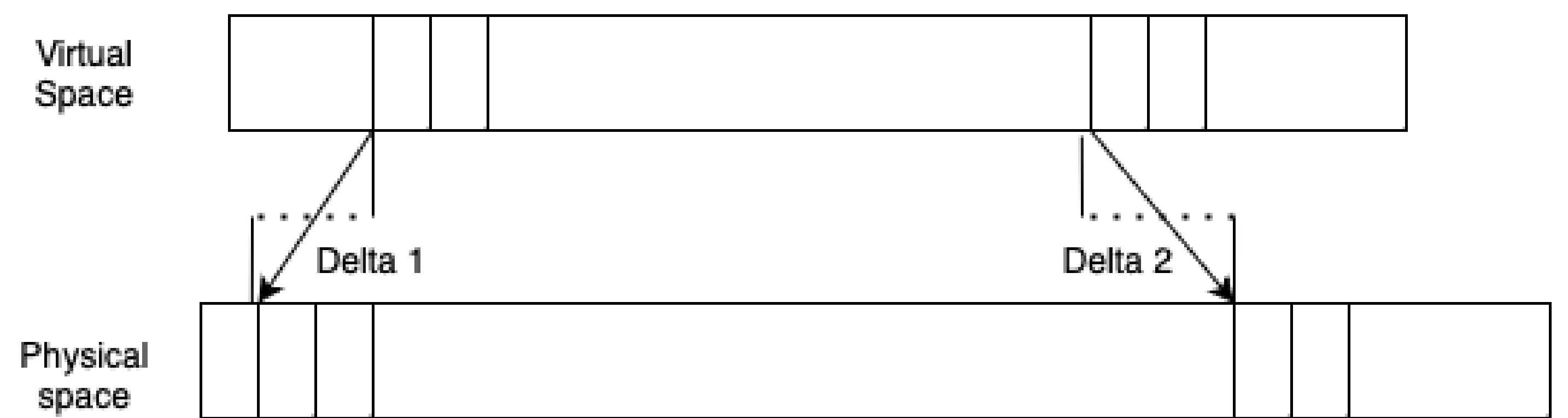
# Expirement Proposal



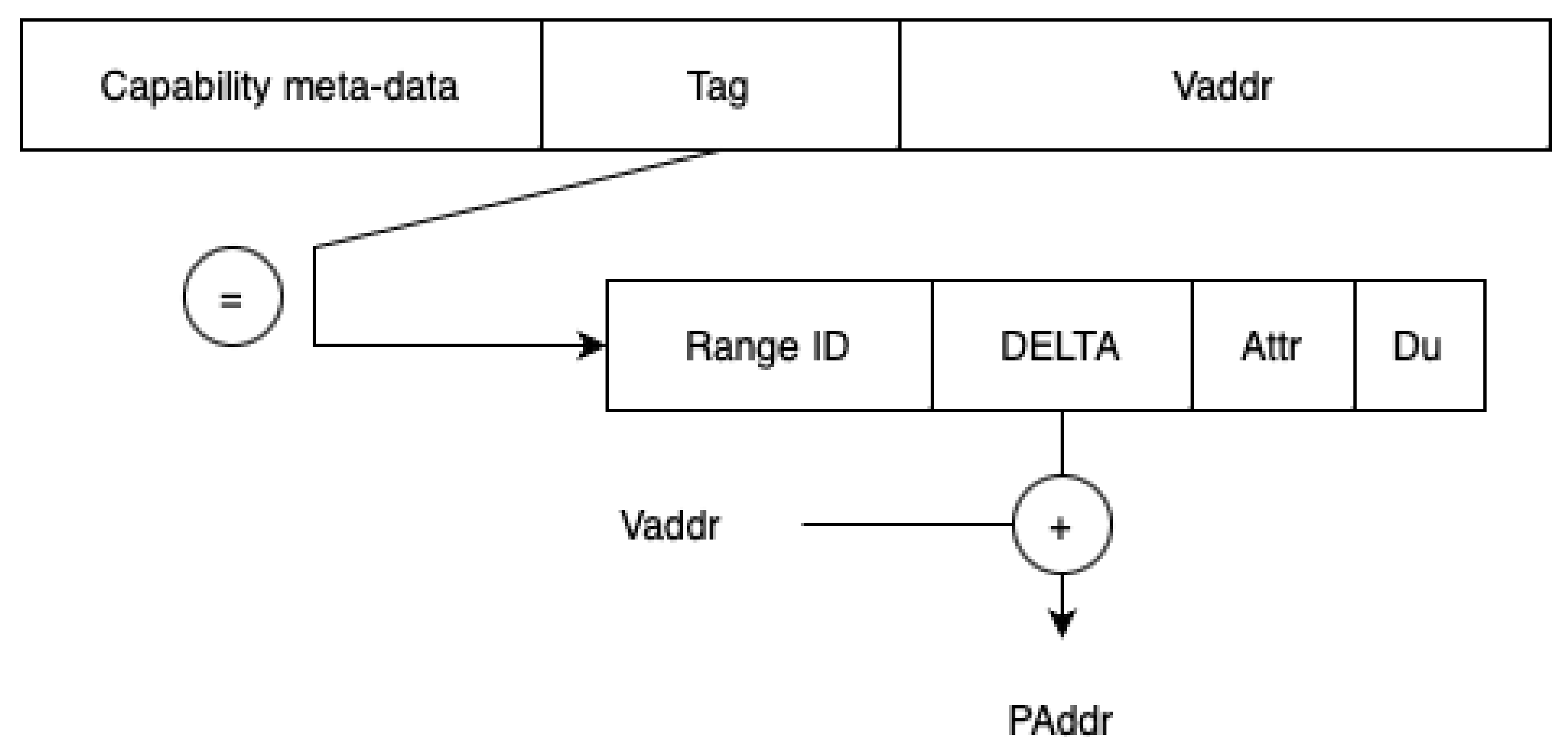
The above figure provides an higher overview of the experiment setup. The experiment is divided into:

- *A set of Microbenchmarks which are designed to create pressure on the L1 TLB to create more L1 misses.*
- *Modified Malloc to call the new syscall mmap\_range for large memory objects.*
- *Testing different memory object sizes to use range TLB.*

## Range TLB



In the current expiriment adopts the range translation concept from FlexPointer which was inturn adopted from RMM. A range consists of contiguous virtual and physical pages. Addresses share a common Delta (physical address - virtual address). To assume physical contiguity of a particular range (We are proposing to use eager paging strategy as proposed from FlexPointer).



Range TLB is organised as a fully associative cache. As per the conventional the page table the lower 12 bits is used record attributes (eg: R/W: Read/Write). Du refers if the current range entry is a dummy one or not (This is used to indentify the next immediate space for a range entry.)

## References